IN THE CLAIMS

Please amend the claims as follows:

1. (Currently Amended) A method [[of]] comprising:

tracking packet sequence numbers of request packets and response packets of transactions transferring data to or from a network interface, said method ~~comprising:~~ including,

for every request packet transmitted by the network interface,

writing the packet sequence number to a location in a circular send queue pointed to by a write pointer and setting a valid bit at said location, wherein the valid bit is indicative of whether at least one response is expected;

incrementing the write pointer if the packet is a read request packet or clearing a read indicator at the location in the circular send queue pointed to by the write pointer if the packet is not a read request packet; and

for every response packet received by the network interface,

checking the packet sequence number of the response packet against the packet sequence number stored at a location in the circular send queue pointed to by the read pointer of the circular send queue.

2. (Original) The method recited in claim 1, wherein a response packet is dropped if the valid bit at the location in the circular send queue pointed to by the read pointer is not set.

3. (Original) The method recited in claim 1, wherein a response packet is accepted if the valid bit at the location in the circular send queue pointed to by the read pointer is set and the packet sequence number of the response packet is both equal to or less than the packet sequence number written at the location in the circular send queue pointed to by the read pointer and greater than the last acknowledged packet sequence number.

4. (Original) The method recited in claim 3, wherein, if the response packet is a read response packet, it is accepted if the packet sequence number of the read response packet is equal

to the packet sequence number written at the location in the circular send queue pointed to by the read pointer and the read indicator is set.

5. (Original) The method recited in claim 3, wherein, if a response packet is accepted, the valid bit at the location in the circular queue pointed to by the read pointer is cleared and, if the response packet is a read response packet, the read pointer is incremented.

6. (Original) The method recited in claim 1, further comprising:

for every request packet received in the network interface,

writing the packet sequence number to a location in a circular receive queue pointed to by a write pointer and setting a valid bit at the location in the circular receive queue pointed to by the write pointer;

if the request packet is a read request packet, then setting the read bit at the location in the circular receive queue pointed to by the write pointer and incrementing the write pointer;

if the request packet is not a read request packet, then clearing the read bit at the location in the circular receive queue pointed to by the write pointer; and

reading the packet sequence number and valid bit at a location pointed to by the read pointer of the circular receive queue.

7. (Original) The method recited in claim 6, wherein a response packet is transmitted if the valid bit at the location pointed to by the read pointer of the circular receive queue is set.

8. (Original) The method recited in claim 7, wherein a read response packet is transmitted if the read bit at the location pointed to by the read pointer of the circular receive queue is set.

9. (Original) The method recited in claim 7, wherein the valid bit at the location of the circular receive queue pointed to by the read pointer is cleared and, if the response packet is a read response packet, the read pointer is incremented after the response packet is transmitted.

10. (Currently Amended) A computer program stored in a memory on a network interface, said program, when executed, causing said network interface to carry out a method ~~of~~ comprising:

tracking packet sequence numbers of request packets and response packets of transactions transferring data to or from said network interface, said method ~~comprising:~~ including,

for every request packet transmitted by the network interface,

writing the packet sequence number to a location in a circular send queue pointed to by a write pointer and setting a valid bit at said location, wherein the valid bit is indicative of whether at least one response is expected;

incrementing the write pointer if the packet is a read request packet or clearing a read indicator at the location in the circular send queue pointed to by the write pointer if the packet is not a read request packet; and

for every response packet received by the network interface,

checking the packet sequence number of the response packet against the packet sequence number stored at a location in the circular send queue pointed to by the read pointer of the circular send queue.

11. (Original) The computer program recited in claim 10, wherein a response packet is dropped if the valid bit at the location in the circular send queue pointed to by the read pointer is not set.

12. (Original) The computer program recited in claim 10, wherein a response packet is accepted if the valid bit at the location in the circular send queue pointed to by the read pointer is set and the packet sequence number of the response packet is both equal to or less than the packet sequence number written at the location in the circular send queue pointed to by the read pointer and greater than the last acknowledged packet sequence number.

13. (Original) The computer program recited in claim 12, wherein, if the response packet is a read response packet, it is accepted if the packet sequence number of the read response packet is

equal to the packet sequence number written at the location in the circular send queue pointed to by the read pointer and the read indicator is set.

14. (Original) The computer program recited in claim 12, wherein, if a response packet is accepted, the valid bit at the location in the circular queue pointed to by the read pointer is cleared and, if the response packet is a read response packet, the read pointer is incremented.

15. (Original) The computer program recited in claim 10, further comprising:

for every request packet received in the network interface,

writing the packet sequence number to a location in a circular receive queue pointed to by a write pointer and setting a valid bit at the location in the circular receive queue pointed to by the write pointer;

if the request packet is a read request packet, then setting the read bit at the location in the circular receive queue pointed to by the write pointer and incrementing the write pointer;

if the request packet is not a read request packet, then clearing the read bit at the location in the circular receive queue pointed to by the write pointer; and

reading the packet sequence number and valid bit at a location pointed to by the read pointer of the circular receive queue.

16. (Original) The computer program recited in claim 15, wherein a response packet is transmitted if the valid bit at the location pointed to by the read pointer of the circular receive queue is set.

17. (Original) The computer program recited in claim 16, wherein a read response packet is transmitted if the read bit at the location pointed to by the read pointer of the circular receive queue is set.

18. (Original) The computer program recited in claim 16, wherein the valid bit at the location of the circular receive queue pointed to by the read pointer is cleared and, if the response is a read response, the read pointer is incremented after the response packet is transmitted.

19. (Currently Amended) A network interface comprising:

a transmitter;

a receiver;

a send queue context memory;

a receive queue context memory;

a send queue engine connected to the send queue context memory, the transmitter and the receiver, wherein the send queue engine is connected to the send queue context memory by a first connection, and the send queue context memory is connected to both the transmitter and the receiver through the first connection and through the send queue engine; and

a receive queue engine partitioned from the send queue engine and connected to the receive queue context memory, the transmitter and the receiver, wherein the receive queue engine is connected to the receive queue context memory by a second connection separate from the first connection, and the receive queue context memory is connected to both the transmitter and the receiver through the second connection and through the receive queue engine.

20. (Original) A network interface according to claim 19, further comprising a plurality of ports receiving data from a corresponding plurality of NGIO or Infiniband serial links.

21. (Original) A network interface according to claim 20, further comprising a virtual interface architecture to establish communication with said plurality of NGIO or Infiniband serial links.

22. (Previously Presented) The network interface according to claim 19, wherein the first connection directly connects the send queue engine to the send queue context memory, and the second connection directly connects the receive queue engine partitioned from the send queue engine to the receive queue context memory.

23. (Previously Presented) The network interface according to claim 19, wherein the send queue engine is directly connected to the transmitter and directly connected to the receiver.

24. (Previously Presented) The network interface according to claim 19, wherein the receive queue engine is directly connected to the transmitter and directly connected to the receiver.